May 04, 2013

iMX233-OLinuXino: Current State

Overview

It has been a while since I played the last time with my OLinuXino, in the meantime Robert Nelson has done a great job in keeping his guide and the build script up to date but I prefer to build everything manually and just take some of his patches.

This post is work in process so print it into an PDF file if you want to keep an older version as reference.

The Kernel

Please have a look at *Building a kernel 3.x for the iMX233-OLinuXino* for more detailed build instructions.

You can use a kernel straight from https://www.kernel.org/:

```
[chris@thinkpad OLinuXino]$ git clone git://git.kernel.org/pub/scm/linux
/kernel/git/stable/linux-stable.git linux-stable.git
[chris@thinkpad OLinuXino]$ cd linux-stable.git/
[chris@thinkpad linux-stable.git]$ git checkout v3.9.8
[chris@thinkpad linux-stable.git]$
```

Note

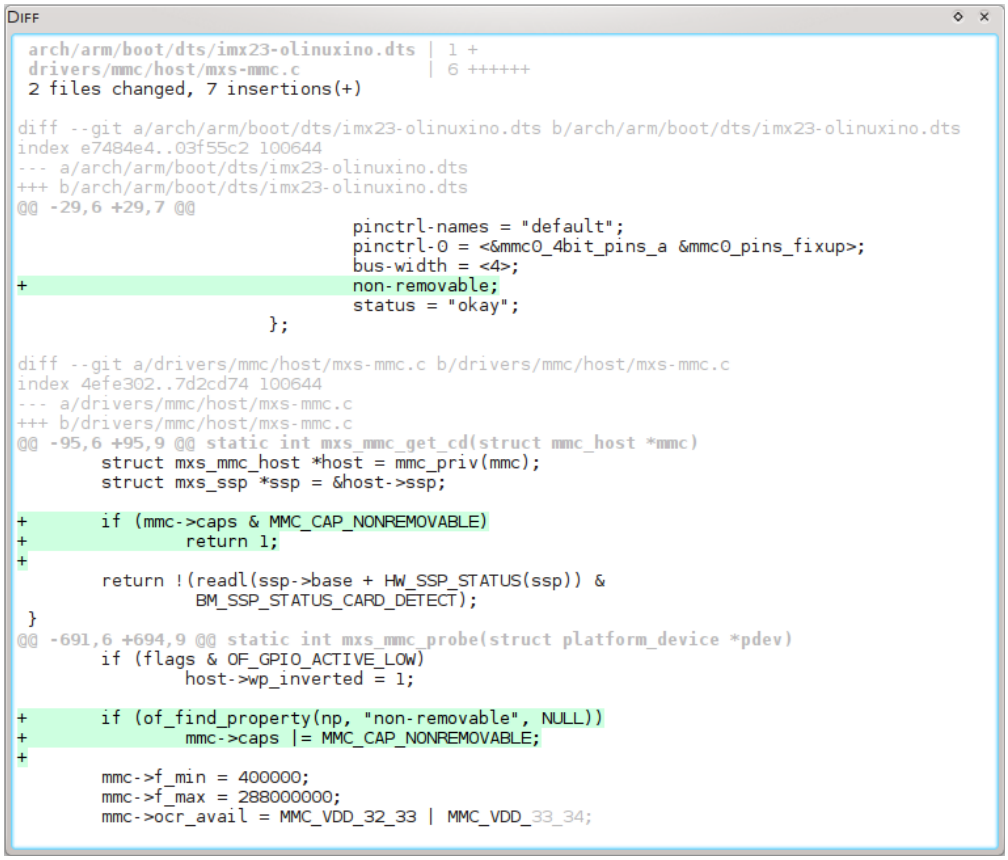
There is also linux-next with some more bleeding edge kernels:

```
[chris@thinkpad OLinuXino]$ git clone git://git.kernel.org/pub/scm/linux
/kernel/git/next/linux-next.git linux-next.git
[chris@thinkpad linux-next.git]$ git checkout v3.10-rc
v3.10-rc1   v3.10-rc2   v3.10-rc3   v3.10-rc4   v3.10-rc5   v3.10-rc6
v3.10-rc7
[chris@thinkpad linux-next.git]$ git checkout v3.10-rc7
[chris@thinkpad linux-next.git]$
```

Now we need to apply two patches that will mark the SD card as non-removable to fix the `Waiting for root device /dev/mmcblk0p2...` problem so that the boot process will not hang:

```
[chris@thinkpad linux-stable.git]$ wget https://github.com/RobertCNelson
/armv5_devel/raw/v3.9.x-imxv5/patches/mxs_mmc/0001-mmc-mxs-mmc-Add-support-
for-non-removable-cards.patch
[chris@thinkpad linux-stable.git]$ wget https://github.com/RobertCNelson
/armv5_devel/raw/v3.9.x-imxv5/patches/mxs_mmc/0002-ARM-dts-imx23-olinuxino-
Set-the-sdcard-as-non-remova.patch
[chris@thinkpad linux-stable.git]$ patch -p1 < 0001-mmc-mxs-mmc-Add-support-
for-non-removable-cards.patch
patching file drivers/mmc/host/mxs-mmc.c
[chris@thinkpad linux-stable.git]$ patch -p1 < 0002-ARM-dts-imx23-olinuxino-
Set-the-sdcard-as-non-remova.patch
patching file arch/arm/boot/dts/imx23-olinuxino.dts
[chris@thinkpad linux-stable.git]$
```

This will add only five lines of code, you can easily view the differences with an application like git-cola:



Patches to add support for non-removable SD cards

It looks like the spi patch from Fadil Berisha (https://github.com/koliqi/imx23-olinuxino) has been merged into mainline, so we only need the i2c patch:

To add support for i2c we need to apply the following patches:

```
[chris@thinkpad linux-stable.git]$ wget https://github.com/koliqi/imx23-
olinuxino/raw/master/kernel/0001-MXS-imx23-olinuxino-Add-i2c-support.patch
[chris@thinkpad linux-stable.git]$ patch -p1 < 0001-MXS-imx23-olinuxino-
Add-i2c-support.patch
patching file arch/arm/boot/dts/imx23-olinuxino.dts
Hunk #1 succeeded at 69 (offset 19 lines).
patching file arch/arm/boot/dts/imx23.dtsi
Hunk #1 succeeded at 292 with fuzz 2 (offset 13 lines).
Hunk #2 succeeded at 383 (offset 13 lines).
Hunk #3 succeeded at 435 (offset 15 lines).
[chris@thinkpad linux-stable.git]$
```

Update: To fix some i2c timeouts we will also need the following patch:

```
[chris@thinkpad linux-stable.git]$ wget https://www.dropbox.com
/s/5t5ewcvtsuh4oba/next20130218_i2c-mxs.c.patch
[chris@thinkpad linux-stable.git]$ patch -p1 < next20130218_i2c-mxs.c.patch
patching file drivers/i2c/busses/i2c-mxs.c
[chris@thinkpad linux-stable.git]$ sed -i 's/fsl,imx28-i2c/fsl,imx23-i2c/g'
arch/arm/boot/dts/imx23.dtsi
[chris@thinkpad linux-stable.git]$
```

Now lets configure and build the kernel:

```
[chris@thinkpad linux-stable.git]$ make ARCH=arm CROSS_COMPILE=arm-
none-eabi- mxs_defconfig
  HOSTCC  scripts/basic/fixdep
  HOSTCC  scripts/kconfig/conf.o
  HOSTCC  scripts/kconfig/zconf.tab.o
  HOSTLD  scripts/kconfig/conf
#
# configuration written to .config
#
[chris@thinkpad linux-stable.git]$
```

Note

You can start with my kernel configuration if you want, then you don't have to change anything when running menuconfig:

```
[chris@thinkpad linux-stable.git]$ curl -L http://sourceforge.net/projects
/janncc/files/olinuxino/kernel/3.9.8/dotconfig/download -o .config
[chris@thinkpad linux-stable.git]$ cp .config dotconfig
```

`make oldconfig` is also worth a try.

```
[chris@thinkpad linux-stable.git]$ make ARCH=arm CROSS_COMPILE=arm-
none-eabi- menuconfig
```

Note

To be sure to built all modules directly into the kernel and not as loadable modules you can execute the following command: `sed -i 's/=m/=y/g' .config`

```
[chris@thinkpad linux-stable.git]$ diff .config dotconfig
1233c1233
< CONFIG_GPIO_GENERIC_PLATFORM=m
---
> CONFIG_GPIO_GENERIC_PLATFORM=y
[chris@thinkpad linux-stable.git]$ sed -i 's/=m/=y/g' .config
[chris@thinkpad linux-stable.git]$ diff .config dotconfig
[chris@thinkpad linux-stable.git]$
```

```
[chris@thinkpad linux-stable.git]$ make ARCH=arm CROSS_COMPILE=arm-
none-eabi- zImage modules
scripts/kconfig/conf --silentoldconfig Kconfig
  WRAP    arch/arm/include/generated/asm/auxvec.h
...
    CC      arch/arm/boot/compressed/fdt.o
    CC      arch/arm/boot/compressed/atags_to_fdt.o
    SHIPPED arch/arm/boot/compressed/lib1funcs.S
    AS      arch/arm/boot/compressed/lib1funcs.o
    SHIPPED arch/arm/boot/compressed/ashldi3.S
    AS      arch/arm/boot/compressed/ashldi3.o
    LD      arch/arm/boot/compressed/vmlinux
    OBJCOPY arch/arm/boot/zImage
    Kernel: arch/arm/boot/zImage is ready
    Building modules, stage 2.
    MODPOST 0 modules
[chris@thinkpad linux-stable.git]$
[chris@thinkpad linux-stable.git]$ make ARCH=arm CROSS_COMPILE=arm-
none-eabi- imx23-olinuxino.dtb
  CC      scripts/mod/devicetable-offsets.s
  GEN     scripts/mod/devicetable-offsets.h
  HOSTCC  scripts/mod/file2alias.o
  HOSTLD  scripts/mod/modpost
  DTC     arch/arm/boot/dts/imx23-olinuxino.dtb
[chris@thinkpad linux-stable.git]$
```

Now you have two files `arch/arm/boot/zImage` (the kernel) and `arch/arm/boot/dts/imx23-olinuxino.dtb` (device tree blob) which is everything you need to run the kernel on the OLinuXino.

Note

If you want to repeat this procedure, start with clean-up:

```
[chris@thinkpad linux-stable.git]$ make ARCH=arm CROSS_COMPILE=arm-
none-eabi- distclean
```

You will find these files under: http://sourceforge.net/projects/janncc/files/olinuxino/kernel/3.9.8/

U-Boot

Please have a look *here*.

Installing

Arch Linux ARM

See here: *A new SD card image for the iMX233-OLinuXino*

U-Boot

```
[chris@thinkpad OLinuXino]$ curl -L http://sourceforge.net/projects/janncc
/files/olinuxino/u-boot/uboot_v2013.04/u-boot.sb/download -o u-boot.sb
[chris@thinkpad OLinuXino]$ curl -L http://sourceforge.net/projects/janncc
/files/olinuxino/u-boot/uboot_v2013.04/MD5SUM.TXT/download -o MD5SUM.TXT
[chris@thinkpad OLinuXino]$ md5sum -c MD5SUM.TXT
u-boot.sb: OK
[chris@thinkpad OLinuXino]$ sudo dd if=u-boot.sb of=/dev/mmcblk0p1 bs=512
seek=4
[chris@thinkpad OLinuXino]$ sudo mount /dev/mmcblk0p2 /mnt/olinuxino/
[chris@thinkpad OLinuXino]$ sudo vim /mnt/olinuxino/boot/uEnv.txt
[chris@thinkpad OLinuXino]$ cat /mnt/olinuxino/boot/uEnv.txt
#These are the default settings for some useful u-boot variables:
#Uncomment to override...

fdt_file=/boot/imx23-olinuxino.dtb
optargs=
mmcroot=/dev/mmcblk0p2 rw
mmcrootfstype=ext4 rootwait fixrtc

[chris@thinkpad u-boot.git]$
```

The Kernel

```
[chris@thinkpad ~]$ cd local/OLinuXino/linux-stable.git/
[chris@thinkpad linux-stable.git]$ sudo cp arch/arm/boot/zImage
/mnt/olinuxino/boot/
[chris@thinkpad linux-stable.git]$ sudo cp arch/arm/boot/dts/imx23-
olinuxino.dtb /mnt/olinuxino/boot/
[chris@thinkpad linux-stable.git]$ sudo umount /mnt/olinuxino
[chris@thinkpad linux-stable.git]$
```

Now it should boot.

```
[  OK  ] Found device /dev/ttyAMA0.
         Starting Serial Getty on ttyAMA0...
[  OK  ] Started Serial Getty on ttyAMA0.
[  OK  ] Reached target Login Prompts.
[  OK  ] Started OpenNTP Daemon.
[  OK  ] Reached target Multi-User System.
[  OK  ] Reached target Graphical Interface.

Arch Linux 3.9.8-dirty (ttyAMA0)

olinuxino login: root
Password:
Last login: Wed Dec 31 18:04:09 on ttyAMA0
[root@olinuxino ~]#
```

I haven’t tested everything yet so if you encounter a problem with the new kernel just leave a comment.

WiFi is working at least:

```
[root@olinuxino ~]#
[ 2301.080000] usb 1-1: new high-speed USB device number 2 using ci_hdrc
[ 2301.270000] usb 1-1: ath9k_htc: Firmware htc_7010.fw requested
[ 2301.460000] usb 1-1: ath9k_htc: Transferred FW: htc_7010.fw, size: 72992
[ 2301.540000] ath9k_htc 1-1:1.0: ath9k_htc: HTC initialized with 45 credits
[ 2301.950000] ath9k_htc 1-1:1.0: ath9k_htc: FW Version: 1.3
[ 2302.000000] ieee80211 phy0: Atheros AR9287 Rev:2
[root@olinuxino ~]# wifi-menu
:: Scanning for networks  [ BUSY ]
:: wlan0-AndroidAP up  [ BUSY ]
Successfully initialized wpa_supplicant
rfkill: Cannot open RFKILL control device
[ 2473.460000] wlan0: authenticate with 02:1a:11:f4:b8:ba
[ 2474.150000] wlan0: send auth to 02:1a:11:f4:b8:ba (try 1/3)
[ 2474.170000] wlan0: authenticated
[ 2474.180000] wlan0: associate with 02:1a:11:f4:b8:ba (try 1/3)
[ 2474.200000] wlan0: RX AssocResp from 02:1a:11:f4:b8:ba (capab=0x411
status=0 aid=2)
[ 2474.230000] wlan0: associated
 [ DONE ]
[root@olinuxino ~]# ping google.de
PING google.de (173.194.41.151) 56(84) bytes of data.
64 bytes from lhr08s03-in-f23.1e100.net (173.194.41.151): icmp_seq=1 ttl=51
time=72.9 ms
64 bytes from lhr08s03-in-f23.1e100.net (173.194.41.151): icmp_seq=2 ttl=51
time=83.3 ms

--- google.de ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 72.937/78.156/83.375/5.219 ms
[root@olinuxino ~]#
```

Testing the Hardware

First we need to have a look at the pinout:

Description	GPIO	CON1					CON2	GPIO	Description
+5V External		5Vex		USB		Video	5Vext		+5V External
Ground		GND					GND		Ground
P1/LCD_D00	32	3	(top side of board)				3.3Vre		+3.3 V Regulated
P2/LCD_D01	33	4					GND		Ground
P3/LCD_D02	34	5	Olimex				5	30	AUART1_TxD
P4/LCD_D03	35	6	Olinuxino-Micro				6	31	AUART1_RxD
P5/LCD_D04	36	7					7	23/55	I2C_SCL
P6/LCD_D05	37	8					8	25/56	I2C_SDA
P7/LCD_D06	38	9					9	0	P9/LCD08/SSP2_MISO
P8/LCD_D07	39	10					10	20	SSP2_MOSI
P9/LCD08/SSP2_MISC	0	11	reset bu				11	24	SSP2_SCK
P10/LCD_D09	1	12					12	3/19	CS_UEXT_GPIO
P11/LCD_D10	2	13					13	26	PWM0/DUART_RxD
P12/LCD_D11	3	14					14	27	PWM1/DUART_TxD
P13/LCDD12	4	15					15		SJTAG_PSW
P14/LCD_D13	5	16					16		HPL
P15/LCD_D14	6	17					17		HPR
P16/LCD_D15	7	18					18		HP_VGND
P17/LCD_D16	16	19	power b				19		LIN1_INL
P18/LCD_D17/USB_EI	17	20					20		LIN1_INR
P19/LCD_DOTCLK	54	21					21		PIN34/MIC
P20/LCD_VSYNC	57	22					22		PIN33/LRADC0
P21/HSYNC/I2C_SDA	56	23					23		PIN32/LRADC1
P22/ENA/I2C_SCL	55	24	recover				24	92	PIN31
P23/LCD_DISP	50	25					25	91	PIN30
P24/LCD_WR	52	26	UART			SJTAG	26	23	PIN29/SOFT_SCL
P25/LCD_RS	51	27	DEBUG			DEBUG	27	25	PIN28/SOFT_SDA
P26/LCD_CS	53	28	GND			SJTAG_PSV	28	60	PIN27/PWM2
Ground		GND	Tx	+5V jack	uSD card	GND	GND		Ground
Open or 3.3V		(3V3	Rx			3V3	BAT		Battery
								RED	vital function, gpio disabled (duart) or altered (usb pull-up)
Ref: Olinuxino User Manual rev.E page 30-31								A/B=	jumper default/alternate
https://www.olimex.com/dev/OLINUXINO/iMX233-OLINUXINO/OLINUXINO-MICRO_revision_E.pdf									

iMX233-OLinuXino-Micro pinout (Red means you might break something else if you use it (usb, debug UART), and blue means that the physical pin is used in multiple places. Source: https://docs.google.com/spreadsheet/ccc?key=0AspkrcYcY5bWdFB6WC1xRlp5bFRjc1hwVnlQZDVmeUE)

GPIO

We will start with something simple like toggling the GPIO pin 4 on Connector 1 pin 15. For this purpose we write a little bash script that first exports the GPIO pin and then toggles it in an endless loop:

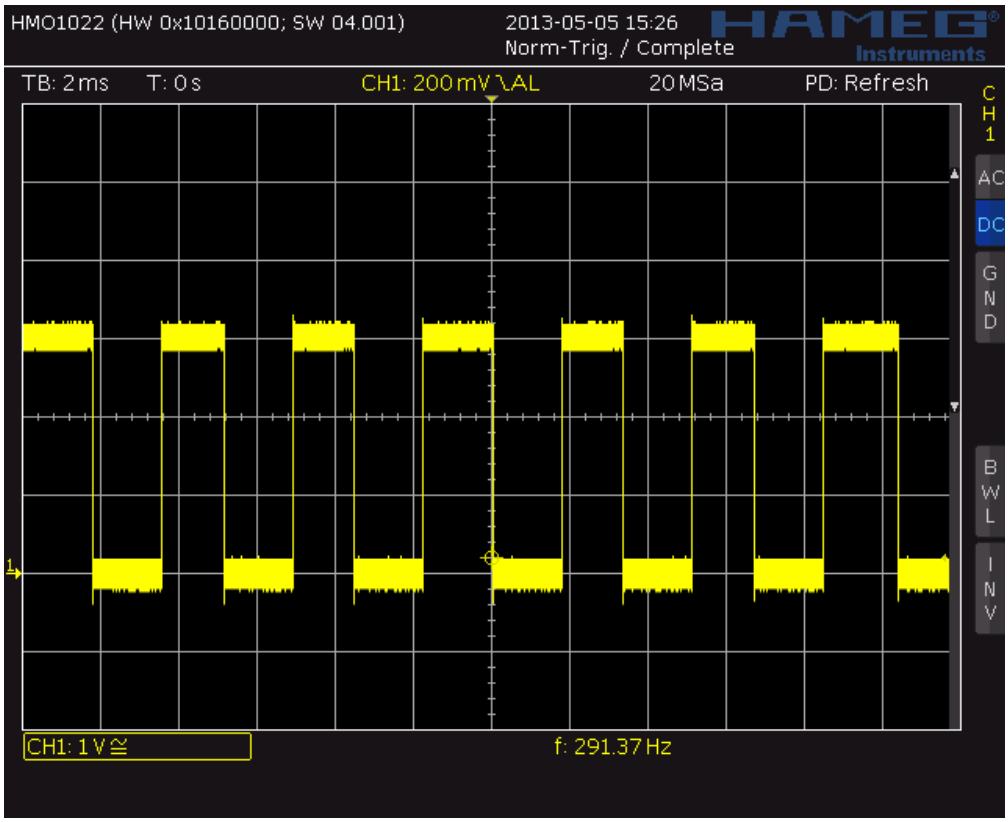
```
[root@olinuxino ~]# vim toggle.sh
[root@olinuxino ~]# cat toggle.sh
#!/bin/bash

echo 4 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio4/direction

# toggle gpio4 with maximum frequency (not memory mapped)
while :
do
  echo 1 > /sys/class/gpio/gpio4/value
  #sleep 1
  echo 0 > /sys/class/gpio/gpio4/value
  #sleep 1
done

[root@olinuxino ~]# chmod +x toggle.sh
[root@olinuxino ~]# ./toggle.sh
```

With this simple script we can toggle the pin with a maximum frequency of about 300 Hz:



iMX233-OLinuXino-Micro maximum toggle speed (bash script)

If we are using memory mapped access we get mush higher toggling speeds of around 5.6 MHz. Have a look here.

```
[root@olinuxino ~]# curl -kL https://github.com/OLIMEX/OLINUXINO/raw/master
/SOFTWARE/iMX233/gpio-mmap.h -o gpio-mmap.h
[root@olinuxino ~]# vim toggle.c
[root@olinuxino ~]# cat toggle.c
```

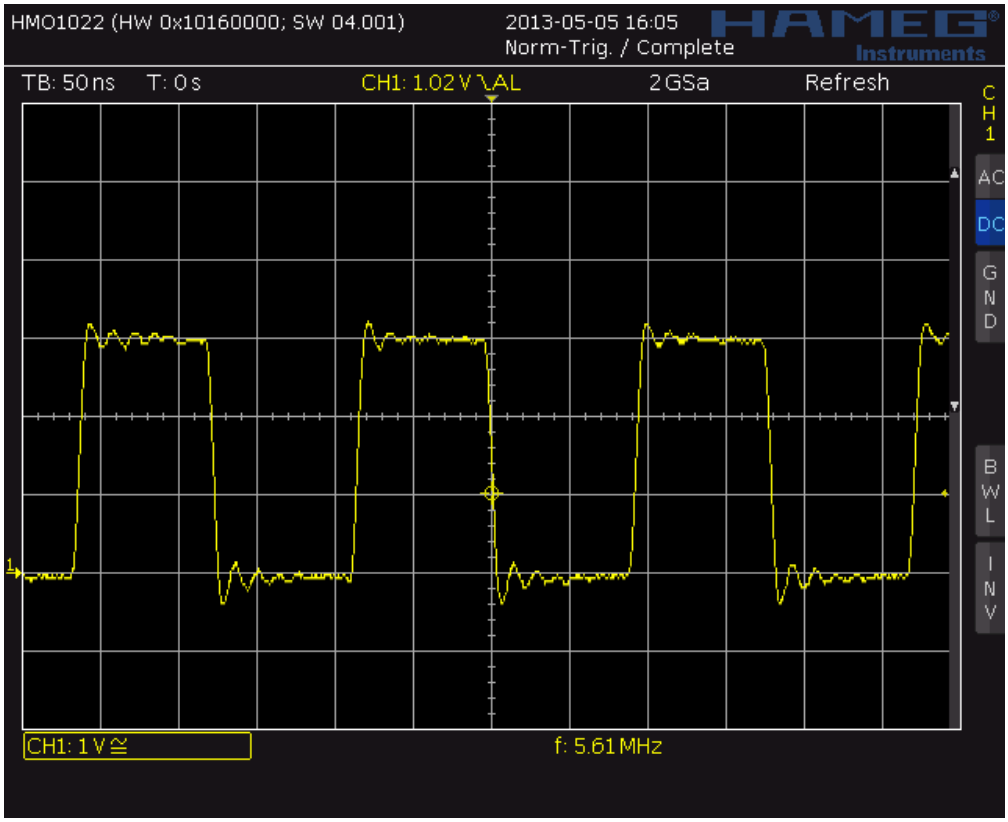
```
#include "gpio-mmap.h"

int main(void)
{
    gpio_map();
    gpio_output(0,4); // bank 0 bit 4 = GPIO4
                      // 4/32 = 0 remainder 4 -> bank 0, pin 4

    while(1)
    {
      GPIO_WRITE_PIN(4,1);
      GPIO_WRITE_PIN(4,0);
    }
}
```

```
[root@olinuxino ~]# gcc toggle.c -o toggle
[root@olinuxino ~]# ./toggle
```

And the result:



iMX233-OLinuXino-Micro maximum toggle speed (memory mapped access)

I2C

It looks good, I have a i2c device:

```
[root@olinuxino ~]# ls /dev/i2*
/dev/i2c-0
[root@olinuxino ~]# pacman -S i2c-tools
[root@olinuxino ~]# i2cdetect -l
i2c-0   i2c             MXS I2C adapter                         I2C adapter
[root@olinuxino ~]# i2cdetect 0
WARNING! This program can confuse your I2C bus, cause data loss and worse!
I will probe file /dev/i2c-0.
I will probe address range 0x03-0x77.
Continue? [Y/n] y
    0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
[root@olinuxino ~]#
```

Now I will try to control Olimex's MOD-IO2 via i2c that I've *already used with the Raspberry Pi*.

First we need to compile i2c-tool which is a simple program that can be used to send some bytes over i2c:

```
[root@olinuxino ~]# curl -kL https://github.com/OLIMEX/OLINUXINO/raw/master
/SOFTWARE/iMX233/I2C/MOD-IO2/i2c-tool.c -o i2c-tool.c
[root@olinuxino ~]# gcc i2c-tool.c -o i2c-tool
[root@olinuxino ~]# ./i2c-tool -w 0 0x48 4 0x02 0xA0 0x40 0x03
SLAVE ADDRESS: 0x48
NUMBER OF BYTES TO WRITE: 4
MEMORY ALLOCATED AT ADDRESS: 0x1D80008
/dev/i2c-0 OPENDED!
Failed writing to the I2C-bus: Connection timed out
[root@olinuxino ~]#
```

So far so good, I can execute the i2c-tool and I hope I get the timeout because nothing is connected to the i2c pins.

Now we have to find the correct pins on the board, i2c support gets added by the patch `0001-MXS-imx23-olinuxino-Add-i2c-support.patch` which adds 3 different pin mappings to imx23.dtsi:

```
Diff
@@ -292,6 +292,39 @@
                                        fsl,voltage = <1>;
                                        fsl,pull-up = <1>;
                                };
+
+                               i2c0_pins_a: i2c0@0 {
+                                       reg = <0>;
+                                       fsl,pinmux-ids = <
+                                               0x01e0 /* MX23_PAD_I2C_SCL__I2C_SCL */
+                                               0x01f0 /* MX23_PAD_I2C_SDA__I2C_SDA */
+                                       >;
+                                       fsl,drive-strength = <1>;
+                                       fsl,voltage = <1>;
+                                       fsl,pull-up = <1>;
+                               };
+
+                               i2c1_pins_a: i2c1@0 {
+                                       reg = <0>;
+                                       fsl,pinmux-ids = <
+                                               0x1171 /* MX23_PAD_LCD_ENABLE__I2C_SCL */
+                                               0x1181 /* MX23_PAD_LCD_HSYNC__I2C_SDA  */
+                                       >;
+                                       fsl,drive-strength = <1>;
+                                       fsl,voltage = <1>;
+                                       fsl,pull-up = <1>;
+                               };
+
+                               i2c2_pins_a: i2c2@0 {
+                                       reg = <0>;
+                                       fsl,pinmux-ids = <
+                                               0x2031 /* MX23_PAD_SSP1_DATA1__I2C_SCL  */
+                                               0x2041 /* MX23_PAD_SSP1_DATA2__I2C_SDA  */
+                                       >;
+                                       fsl,drive-strength = <1>;
+                                       fsl,voltage = <1>;
+                                       fsl,pull-up = <1>;
+                               };
                        };

                        digctl@8001c000 {
@@ -350,7 +383,7 @@
                                reg = <0x80038000 0x2000>;
                                status = "disabled";
                        };
```

iMX233-OLinuXino i2c pin mappings patch

from which `i2c1_pins_a` (`MX23_PAD_LCD_ENABLE__I2C_SCL`, `MX23_PAD_LCD_HSYNC__I2C_SDA`) gets selected in the device tree (imx23-olinuxino.dts):

```
Diff
@@ -69,6 +69,12 @@
                };

                apbx@80040000 {
+                       i2c0: i2c@80058000 {
+                               pinctrl-names = "default";
+                               pinctrl-0 = <&i2c1_pins_a>;
+                               status = "okay";
+                       };
+
                        duart: serial@80070000 {
                                pinctrl-names = "default";
                                pinctrl-0 = <&duart_pins_a>;
```

iMX233-OLinuXino i2c device tree

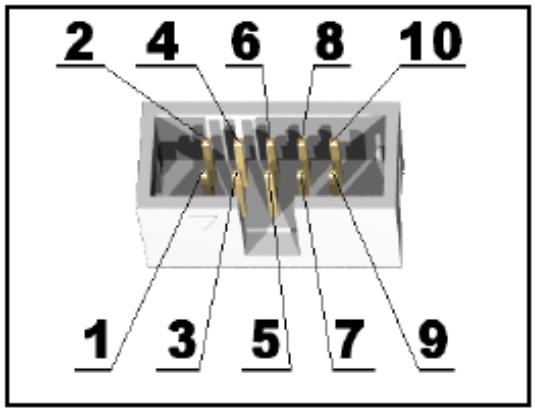
Now if we look into the iMX233 Reference Manual (17,5 MB) and search for `LCD_ENABLE` we will find *Table 36-3. 128-Pin LQFP Pin Definitions by Pin Number* which tells us that `LCD_ENABLE` is connected to processor pin 11 and `LCD_HSYNC` is connected to processor pin 15 and both have i2c as alternate function.

After looking into the OLinuXino-MICRO USER'S MANUAL we know that processor pin 11 (**I2C_SCL**) is available at **CON1.24** and processor pin 15 (**I2C_SDA**) was routed to **CON1.23**.

Since the MOD-IO2 uses a UEXT connector have a look at the UEXT pinout specification:

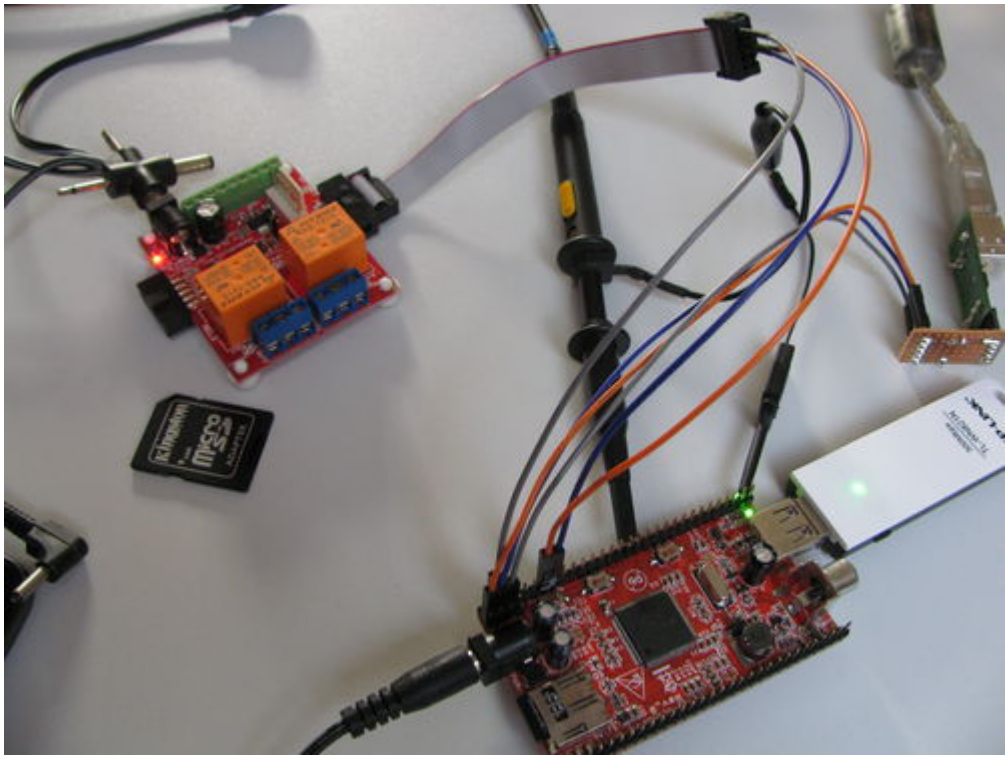
Connector pins description:

Pin #	Signal Name
1	3.3V
2	GND
3	TXD
4	RXD
5	SCL
6	SDA
7	MISO
8	MOSI
9	SCK
10	SSEL



UEXT pinout specification

Now connect SDA with SDA, SCL with SCL and GND with GND:



iMX233-OLinuXino-Micro and MOD-IO2

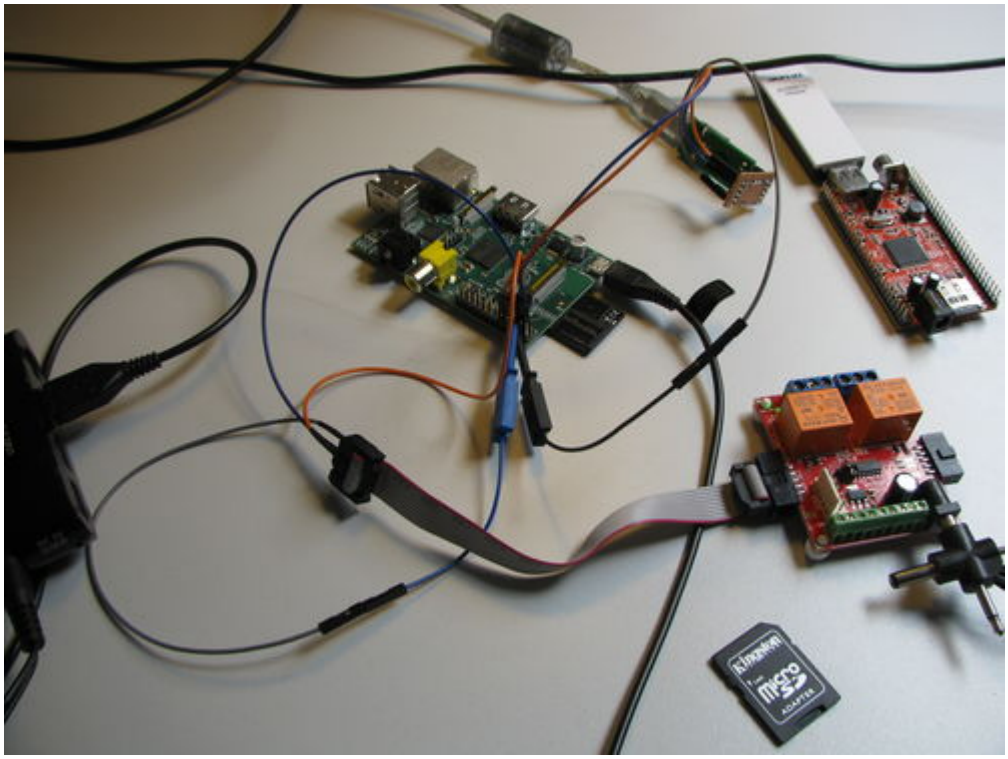
But i2cdetect still does not detect the MOD-IO2 (*unlike on the Raspberry Pi* where it finds it at address `0x48`) and I get timeouts when writing something to the i2c device:

```
[root@olinuxino ~]# i2cdetect 0
WARNING! This program can confuse your I2C bus, cause data loss and worse!
I will probe file /dev/i2c-0.
I will probe address range 0x03-0x77.
Continue? [Y/n] y
    0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
[root@olinuxino ~]# ./i2c-tool -w 0 0x48 4 0x02 0xA0 0x40 0x03
SLAVE ADDRESS: 0x48
NUMBER OF BYTES TO WRITE: 4
MEMORY ALLOCATED AT ADDRESS: 0x99F008
/dev/i2c-0 OPENDED!
Failed writing to the I2C-bus: Connection timed out
[root@olinuxino ~]#  echo "test" > /dev/i2c-0
-bash: echo: write error: Connection timed out
[root@olinuxino ~]#
```

It should look like this:

```
[root@alarmpi ~]# i2cdetect 0
WARNING! This program can confuse your I2C bus, cause data loss and worse!
I will probe file /dev/i2c-0.
I will probe address range 0x03-0x77.
Continue? [Y/n] y
    0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- 48 -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
[root@alarmpi ~]# i2c-tool -w 0 0x48 4 0x02 0xA0 0x40 0x01 #REL 1 on
SLAVE ADDRESS: 0x48
NUMBER OF BYTES TO WRITE: 4
MEMORY ALLOCATED AT ADDRESS: 0x12A3008
/dev/i2c-0 OPENDED!
WRITE:SUCCESS
[root@alarmpi ~]#
```

And since I've tried it again on the Raspberry Pi, I'm sure everything is connected correctly (?) and the hardware is OK:



MOD-IO2 on Raspberry Pi

Its (partly) working now after adding a second i2c patch (`next20130218_i2c-mxs.c.patch`) from Nicolas Le Falher, who was so kind to add a comment below this blog post.

Then I've changed `compatible = "fsl,imx28-i2c"` to `compatible = "fsl,imx23-i2c"` via sed in `arch/arm/boot/dts/imx23.dtsi`.

It is also necessary to use the option `-r` in i2cdetect, else it will show nothing (it is also a lot slower than on the Raspberry Pi where i2cdetect responds instantly).

```
[root@olinuxino ~]# i2cdetect -r 0
WARNING! This program can confuse your I2C bus, cause data loss and worse!
I will probe file /dev/i2c-0 using read byte commands.
I will probe address range 0x03-0x77.
Continue? [Y/n] y
    0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- 48 -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
[root@olinuxino ~]#
```

But it is not possible to switch the relays on or off even though i2c-tool reports success:

```
[root@olinuxino ~]# ./i2c-tool -w 0 0x48 4 0x02 0xA0 0x40 0x03 #REL 1&2 on
SLAVE ADDRESS: 0x48
NUMBER OF BYTES TO WRITE: 4
MEMORY ALLOCATED AT ADDRESS: 0x1F15008
/dev/i2c-0 OPENDED!
WRITE:SUCCESS
[root@olinuxino ~]#
```

I've created a diff of all my changes after I've staged the modified files in git (all_changes_3.9.8.diff):

```
[chris@thinkpad linux-stable.git]$ git add -u
[chris@thinkpad linux-stable.git]$ git diff --cached > all_changes_3.9.diff
[chris@thinkpad linux-stable.git]$ git status
# HEAD detached at v3.9.8
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       modified:   arch/arm/boot/dts/imx23-olinuxino.dts
#       modified:   arch/arm/boot/dts/imx23.dtsi
#       modified:   drivers/i2c/busses/i2c-mxs.c
#       modified:   drivers/mmc/host/mxs-mmc.c
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#       all_changes_3.9.8.diff
#       dotconfig
[chris@thinkpad linux-stable.git]$
```

To be continued.

SPI

I can't find anything about spi under `/dev`, at least dmesg shows something:

```
[root@olinuxino ~]# ls /dev/ |grep spi
[root@olinuxino ~]# dmesg | grep spi
[    1.150000] mxs-spi 80034000.ssp: registered master spi32766 (dynamic)
[root@olinuxino ~]#
```

- https://www.kernel.org/doc/Documentation/spi/spidev_test.c

To be continued.

References/Further Reading

- I2C and the OLinuXino: https://www.olimex.com/forum/index.php?topic=16.0, https://www.olimex.com/forum/index.php?topic=283.0

Last updated: 2013-05-30

Posted by Christian Jann

Filed under: English

Tags: Embedded Linux, OLinuXino, iMX233, ARM, HOWTO, U-Boot, Arch Linux ARM, Linux Kernel, UEXT